BSTZ No. 42390P11770
Express Mail No. EL802873127US

UNITED STATES PATENT APPLICATION

FOR

TIME VARYING PRESENTATION OF ITEMS BASED ON A KEY HASH

Inventors:

Carl M. Ellison
Stephen H. Dohrmann

Prepared by:

Blakely, Sokoloff, Taylor & Zafman LLP
12400 Wilshire Boulevard, Suite 700
Los Angeles, California 90025
(714) 557-3800

TIME VARYING PRESENTATION OF ITEMS BASED ON A KEY HASH

FIELD

This invention relates to the field of data security. In particular, the invention relates to a key verification technique.

GENERAL BACKGROUND

As the number of electronic resources proliferate, the demand for applications to facilitate communications between such resources will also increase. Such applications can include electronic commerce but also secure sharing of data through encrypted or digitally signed electronic mail (e-mail) or secure access to resources through file sharing or remote computer log-on. All of these uses of cryptography require the authentication of users and other data prior to performance of a particular action. That is, it requires a level of trust to be established prior to performance of the transaction.

Specifically, in this networking age, a person is normally authenticated not by personal appearance but rather by use of a cryptographic key. In order for a particular key to correctly represent some person (or other entity), the party accepting the key would need to establish that the particular key is controlled by that person or entity. This process is normally referred to as "key verification". Current key verification techniques are either inadequate and therefore prone to error or abuse or they are so unusual and technical that an average human user might shy away from that activity.

For instance, one key verification technique (referred to as "directory verification" and first described in an Institute of Electronic and Electrical Engineering (IEEE) Transaction on Information Theory publication entitled "New Directions in Cryptography" by Whitfield Diffie and Martin Hellman (November 1976, pp. 644-

654)) involves the publication and global distribution of a printed reference that includes the name, address and assigned public key for each user. The reference is published by a trusted source and distributed in a secure manner. One of many disadvantages of the directory verification technique is that it is costly to implement. Namely, this technique would incur additional costs for publication of the reference, secure distribution of the reference, and for each listed keyholder in the directory, the proof to the trusted source that the user is the true owner of the public key prior to publication. Another disadvantage is that the names of the users may be unique in a small group, but such uniqueness diminishes for larger groups. Hence, as the size of the group gradually expands, name collisions are almost certain to occur. When names collide, any party relying on the directory's results will not always be able to locate the correct directory entry for a desired person or other entity with certainty and therefore be not always able to locate that person's or entity's public key.

Other key verification techniques include the exchange of a digital certificate in accordance with, for example, Request For Comment (RFC-2459) entitled "Internet X.509 Public Key Infrastructure" (January 1999). However, the X.509 mechanism also suffers from the requirement of a central trusted source and increased costs for establishing such a certificate mechanism. It also suffers from name collision with the added disadvantage that when names collide, the user of a certificate may not be aware of the collision because he or she sees only the one directory line item contained within the certificate at hand and not the neighboring region of the directory.

Yet another key verification technique has been established by the application program referred to as "Pretty Good Privacy" (PGP). The idea of PGP key verification is to bind a public key of an owner to his or her global name, such as an e-mail address, for example, without the cost of a central trusted source. PGP allows every user to

generate his or her own keys and certificates. For key verification purposes, PGP computes a fingerprint of the key to be verified, in the form of a cryptographic hash result of that key. This hash result is computed independently by the keyholder's copy of PGP and the relying party's copy of PGP. These cryptographic hash results, displayed either in the form of a long hexadecimal number or a sequence of English words, are then compared by having one party read the value to the other. If the values match, then the key in question is verified.

The PGP key verification technique has the disadvantage that the technique of visually or audibly comparing a number of hexadecimal character values or a string of meaningless words is quite time consuming and strange for the user who wants to achieve appropriate authentication levels. Because of that workload, some users skip the verification step entirely.

BRIEF DESCRIPTION OF THE DRAWINGS

The features and advantages of the invention will become apparent from the following detailed description in which:

Figure 1 is an exemplary embodiment of two parties that are performing operations in accordance with one embodiment of the key verification technique.

Figure 2 is an exemplary embodiment of a computing unit in which one embodiment of the invention can be practiced.

Figure 3 is an exemplary embodiment of a data structure for a verification packet of the key verification technique practiced in Figure 1.

Figure 4 is an exemplary embodiment of a flowchart outlining the operations performed in accordance with the key verification technique using a time varying item presentation.

Figure 5 is a first exemplary embodiment of operations for selecting an item based on bit values obtained from the key hash result.

Figure 6 is a second exemplary embodiment of operations for selecting an item based on bit values obtained from the key hash result.

Figure 7 is an exemplary embodiment of operations for generating an item based on bit values obtained from the key hash result.

DETAILED DESCRIPTION

The invention relates to a computing unit and method for key verification through time varying item presentation based on a key hash result. Such time varying item presentation may include (1) successive selection/generation and graphical display of selected images or characters, (2) successive selection/generation and play back of audible sound(s) such as musical note(s), pronounceable syllables, or (3) any other sorts of periodic sensory presentations. In one embodiment, a source (first computing unit) is configured to transmit at least a global identifier and a current local time realized at the source. These parameters (or related variations thereof) may undergo a periodic cryptographic hash function at a destination (second computing unit) to produce a time varying key hash result. The same process is performed at the source. The periodic key hash result is used at both the source and the destination to periodically select or generate items for presentation that are substantially contemporaneous. An observer who is engaged in key verification then senses both the source and the destination simultaneously and can determine by the apparent simultaneity of these time-varying items produced by each that the global identifiers (typically keys) at the source and destination are the same. The longer the observer monitors these time varying items, the more certain he or she is that the two keys are identical.

Herein, certain terminology is used to describe certain features of the invention. For example, a "computing unit" may generally be considered as hardware, software, firmware or any combination thereof that is configured to process information and enable items to be presented to and perceived by the user. Some illustrative examples of a computing unit include a computer (e.g., laptop, hand held, etc.), a wireless telephone handset, alphanumeric pager or any other portable communication device.

When the computing unit is employed as software, such software features a plurality of software modules, each being instructions or code that, when executed, perform certain function or functions. The software is stored in platform readable medium, which is any medium that can store or transfer information. Examples of "platform readable medium" include, but are not limited or restricted to a programmable electronic circuit, a semiconductor memory device, a volatile memory (e.g., random access memory, etc.), a non-volatile memory (e.g., read-only memory, flash memory, etc.), a floppy diskette, a compact disk, an optical disk, a hard drive disk, or any type of link (defined below).

In addition, a "packet" is generally considered herein as a collection of data in a selected format. The packet may be configured as a data stream having a varying bit length or bit segment of a predetermined length. A "key" is normally an encoding and/or decoding parameter. One type of key is a "public key" that need not be kept secret and therefore may also be used to identify a computing unit or its user. The cryptographic hash result of a key (either public or symmetric/secret), assuming the hash is noninvertible and non-colliding, can also be used as an identifier for a computing unit or its user. The term "contemporaneous" means at the same time or generally about the same time with a nominal latency (e.g., less than one second).

Referring now to Figure 1, an exemplary embodiment of two parties that are performing operations in accordance with one embodiment of the key verification technique is shown. Herein, a first user (sender) 100 is in close physical proximity to a second user (receiver) 140. This allows a computing unit 110 of the sender 100 to communicate with a computing unit 120 of the receiver 140 over a link 150. As shown, the link 150 is any communication pathway over a wired/wireless information-carrying medium (e.g., electrical wire, optical fiber, cable, bus, radio frequency "RF", infrared

"IR" or another wireless communication scheme such as Bluetooth™, past or future Institute of Electrical and Electronics Engineers (IEEE) 802.11 standards published November 16, 1998 and entitled "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications" or any future related standards.

As generally shown, the first computing unit 110 uniquely stores data that is used to identify itself or the sender 100. This identification data is referred to as a "global identifier" 115. In one embodiment, a cryptographic public key (PUK1) is just one type of global identifier. Similarly, the second computing unit 120 is configured to store a global identifier 125 that differs from global identifier 115, such as a different cryptographic public key (PUK2) for example.

The first computing unit 110 is capable of broadcasting a verification packet including its global identifier (e.g., PUK1) to all other computing units that are able to receive the broadcast information. When the broadcast is conducted over a wireless medium, all computing units within a specific geographic area 160 and tuned into a certain frequency may receive the verification packet from the first computing unit 110. However, when the broadcast is conducted over a wired medium, all computing units coupled directly or indirectly to the wired medium may receive the verification packet from the first computing unit 110. In the situation where the wired medium is the Internet, any computing unit having access to the Internet may receive the verification packet.

Referring now to Figure 2, an illustrative embodiment of one of the computing units (e.g., computing unit 110) is shown. For illustrative purposes, the computing unit 110 comprises an input/output (I/O) interface 200, internal circuitry 210, a display screen 220 and a keypad 230 integrated into a casing 240. The casing 240 is made of

an inflexible material such as hardened plastic, and thus, protects the internal circuitry 210 from damage and contaminants.

More specifically, the I/O interface 200 enables the reception of incoming data and the transmission of outgoing data. In one embodiment, as shown, the I/O interface 200 may be implemented as an antenna and perhaps transceiver logic for transmitting and/or receiving verification packets as RF-based signals. Of course, other embodiments of the I/O interface 200 may include, but are not limited or restricted to a wired or wireless modem logic, a light emitting diode (LED) transmitter and/or receiver to transfer data through light pulses, and the like. As shown, the internal circuitry 210 controls the I/O interface 200 and the display screen 220 in response to incoming data from the I/O interface 200 and/or the keypad 230. For instance, the internal circuitry 210 may be used to adjust time displacement to cancel any perceived delay between presentation of identical items at computing units 110 and 120 to achieve exact simultaneity. Examples of the internal circuitry 210 include one or more of the following: processor (e.g., microprocessor, application specific integrated circuit, digital signal processor, or micro-controller), memory (nonvolatile or volatile), combinatorial logic, clocking circuit and the like.

As shown, the display screen 220 is a flat panel screen (e.g., liquid crystal display) although any type of display screen may be used. While the display screen 220 may be used as an output device in one embodiment, it is contemplated that the display 220 may be implemented as a touch screen display, thereby operating as an I/O device. For that embodiment, the keypad 230 may be removed. Alternatively, it is contemplated that the computing unit 110 may be implemented with any mechanism or combination of mechanisms that would allow persons to sense time-varying item

presentation. For instance, although the computing unit is shown with a display screen, it is contemplated that the computing unit may be implemented with speakers to provide an audio interface in addition to or in lieu of the display screen. This would allow presentation of time-varying audible sounds. Similarly, the computing unit may be implemented with a tactile device to allow one to compare time-varying patterns by placement of a hand on both computing units.

Referring now to Figure 3, an exemplary embodiment of a data structure for a verification packet 300 of the key verification technique is illustrated. As shown for this embodiment, the verification packet 300 is transmitted from the first computing unit. Herein, the verification packet 300 includes a plurality of fields; namely, an identifier field 310 and a time field 320. Other optional fields may include, but are not limited or restricted to a data field 330.

Herein, the identifier field 310 includes a global identifier for the source; namely, the sender or the first computing unit used by the sender. In one embodiment, the global identifier may be a public key corresponding to a private key held and controlled by the sender or his/her computing unit. For this exemplary embodiment, the identifier field 310 may include PUK1.

The time field 320 includes a value such as the time at which the verification packet 300 is formed by the first computing unit. As subsequent verification packets are formed and transmitted, the time field 320 of those packets will have different values. The data field 330 includes information that is to be transferred between computing units. Examples of such information may include, but are not limited or restricted to software (e.g., application, applet or any segment of code), a table of items

(e.g., images, bit patterns, data representative of audible sound patterns, etc.) or any data to assist in the presentation of time-varying items.

Referring now to Figure 4, an exemplary embodiment of a flowchart outlining the operations performed in accordance with the key verification technique using a time varying item presentation is shown. First, initialization operations are performed prior to transmission of the verification packet from a source to a destination. One initialization operation involves placement of a global identifier associated with the source in the verification packet (see block 400). Another initialization operation involves the selection of a time interval (TI) for updating the key hash result by the sender and the receiver (see block 410). The selection may be accomplished through a prior agreement (e.g., hard coded into software running this application) or by inclusion of the time interval with the verification packet (e.g. part of the data field). Selected in seconds or fractions thereof, this time interval determines the period at which an item is presented by the computing unit. For example, the time interval may determine when another image is illustrated on the display of the computing units, when an audible sound is played back from speakers integrated within or coupled to the computing units and the like.

At blocks 420 and 430, the verification packet is transmitted to the second computing unit, which computes a clock skew between these computing units. In particular, for one embodiment, the clock skew may be computed by the second computing unit recording the time upon which the transmitted verification packet is received and determining a difference between this receipt time and a local time realized at the first computing unit when the verification packet is being formed (hereinafter referred to as the "original source time"). The original source time is

contained in the time field of the verification packet. After computing the clock skew, as shown in block 440, the second computing unit can estimate, within a small error range corresponding to the time it took to deliver the verification packet from the source to the destination, a current local time at the source (T_{source}) as it corresponds to the local time at the second computing unit.

At every time interval (TI), namely when the current source local time (T_{source}) equals zero modulo TI (e.g., T_{source} mod TI = 0), both (T_{source}) and the global identifier for the source (ID_s) undergo a cryptographic hash operation at both the first computing unit and the second computing unit. The result of the cryptographic hash operation produces a key hash result (see blocks 450 and 460). For one embodiment, the key hash result (h_t) is produced by a random cryptographic function "H(x,y)" as set forth below in equation (1).

(1) $h_t = H(ID_s, T_{source})$, where

"ID_s is a global identifier of the source (e.g., a public key), and

"T_{source}" is the current local time at the source at the beginning of the current time interval.

For this embodiment, a truly random source is used to define the mapping from each element of a two-dimensional domain of the function to a value in the range of that function. However, for H(x,y) to be truly random, an extremely large table would be required, which would be difficult to initialize, much less store.

For another embodiment, instead of using a random cryptographic function (H(x,y)), a computational approximation of H(x,y) is performed. For example, there are well-known and recognized cryptographic hash functions such as a Federal Information Processing Standard Publication 180-1 entitled "Secure Hash Standard"

(April 17, 1995), which specifies Secure Hash Algorithm (SHA-1). One type of approximation S(z) is set forth below in equation (2).

$$(2)\ h_t = H(ID, T_{source}) = S(z) = S(T_{source} \,||\, ID_s \,||\, T_{source}),$$ where

"||" denotes a concatenation operation.

In essence, the approximation may be accomplished by using the current source local time (T_{source}) and combining such information with the source global identifier (ID_s) extracted from the identifier field of the verification packet. The "combining" operation may be accomplished through concatenation as set forth in equation (2) or perhaps through other arithmetic or logical operations. The key hash result h_t is a time sequence of apparently random quantities, due to the characteristics of the random function H(x,y) or the approximation of it using S(z) or some other computation involving a cryptographic hash of the two values or some function(s) of those values.

At block 470, the key hash result h_t is used by both the first computing unit and the second computing unit to select an item to be contemporaneously presented to the users of these devices. For one embodiment, an item may be selected from a table known in advance to all parties or from a table transmitted by the sender. Alternatively, the item may be computed such as through a fractal pattern or via some program producing a graphical image or audible sounds. The program that computes these items would be known to all parties in advance or can be transmitted by the first computing unit within the data field of the verification packet. The presentation of an item includes display of an image on a display screen such as within a special dialog box or adjacent to the name of the user, playback of one or more audible sounds, and the like.

These items are presented to the users and compared (block 480). If the items match (i.e., a successful comparison), if desired, another comparison at the next time interval is conducted by a user to determine whether the items have changed contemporaneously and are also matching (block 490). It is important to note that any comparison is merely a brief recognition that two items are displayed generally contemporaneously and are the same or different. If both conditions are repeatedly satisfied as required by the user, the global identifier is verified.

When the global identifier is a public key, the receiver or verifier of that key might use the global identifier to generate a digital certificate or a local database record that binds the key to information about the keyholder (sender). Such subsequent use might include the generation of: an X.509 Distinguished Name certificate, a PGP e-mail name certificate, a Simple Distributed Security Infrastructure (SDSI) or Simple Public Key Infrastructure (SPKI/SDSI) local name certificate, an X9.59 bank account public key record and the like. Thus, for future uses of the public key, the receiver or verifier could use the generated certificate or record for subsequent key verifications without requiring a physical presence and sensing of a time-varying presentation. If there are no future uses of this verified key, the receiver or verifier need not do any such binding and can instead use the verified key for some immediate purpose.

Referring to Figure 5, an exemplary embodiment of operations for selecting an item based on bit values obtained from the key hash result is shown. The value of the key hash result 500 is produced by a cryptographic hash operation on at least some unique data 510 (e.g., a global identifier) and a time varying data 520 (e.g., source local time). Normally, the key hash result 500 contains 128 or more bits. It is contemplated, however, that certain key hash results may have more than 128-bits, such as 160-bits.

Not all of these bits would be needed to select or compute an item. Rather, each key hash result is reduced to some manageable size, such as M bits for example (where $M \leq 32$).

As shown in Figure 5, such reduction can be accomplished by a selection of a selected sub-field 530 of the key hash result 500 for use in accessing an entry of a table 540. As shown, the table 540 features 2^M entries. Alternatively, the reduction can be achieved by separating the key hash result 500 into a number of fields 600 and performing logical operations on bits of these fields 600 to generate an address for accessing data within certain entries of the table 540. For example, one type of logical operation is an Exclusive OR (XOR) 610 as shown in Figure 6.

For example, when indexing a table of 256 items using an 8-bit key reduced hash result, at each interval, a new item may be accessed from the table for display as shown by pseudo-code set forth below in Table A. The probability of an interloper interjecting an unauthorized global identifier that would show the same item as the authorized global identifier is 1/256. However, because all of the key hash results are independent of one another, the probability of the interloper being able to mimic two successive items is 1/65,536. The probability continues to shrink geometrically over time as more and more items are perceived contemporaneously. Thus, the level of security achieved by the user depends on the amount of time comparing items contemporaneously presented at the exchanging computing units. If the time interval between display changes is short, then the overall time to achieve cryptographically strong comparison (in excess of 90 bits) is also short. In an ideal embodiment, that time to achieve strong comparison would be less than or equal to the time a human

would normally spend in looking at or listening to the sample sequence without spending special effort on the task.

TABLE A

```
{ Time Varying Hash pseudocode (in the manner of PASCAL } }
{ This assumes 256 different icons to be displayed, every 3 seconds. }

        const INT = 3; { time interval in seconds between new displays }

        var
            dt  :       integer; {delta time between my time and the remote time}
            id  :       array[] of byte; {global identifier -- probably a public key}
            idlth       :       integer ;  {size of the global ID array}

{ Display icon number n, where n is 8 bits }

        procedure display_icon (n  :   integer)  ; external  ;

{ Call this procedure, display_next, every INT seconds, when (t mod INT = 0). }

        procedure display_next  ;
            var
                t       :       integer  ;      {time on the remote machine}
                shae    :       sha_1_env  ;    {SHA-1 environment state}
                res     :       byte  ;         { the result byte  }
                i       :       integer  ;      { loop variable  }
                hv      :       array [1..20] of byte  ;        {key hash result }
            begin
                t       := time_in_seconds - dt  ;      {convert destination time to source
time}
                t       := t  - (t mod INT)  ;          {back up to the start of the current
interval}
                sha_init ( shae  ) ;                    {initialize the environment state}
                sha_accum_int  ( shae,  t  ) ;          {accumulate an integer's bytes}
                sha_accum_bytes ( shae, id, idlth ) ; { accumulate the ID  }
                sha_accum_int  ( shae,  t  ) ;          {accumulate the time bytes again
}
                sha_final ( shae,  hv ) ;               {get the key hash result }
                res     := 0  ;                         { reduce the key hash result to 1
byte }
                for i   := 1 to 20 do res := res xor hv[i]  ;
                display_icon ( res  ) ;
            end  ;      {display_next  }
```

Referring now to Figure 7, as described in Figures 5 and 6, the key hash result can be reduced by selection of a sub-field or through logical operations (see blocks

700-720). However, in lieu of using a lookup table for selections of items to be presented to the user, the contents of the reduced key hash result are used to generate the item to be presented (block 730). Namely, bits of the reduced key hash result are used to govern generation of the particular item.

For instance, in one embodiment, the key verification technique may be used to generate different types of images. For the M-bits of the reduced key hash result, at least one bit may be reserved to indicate whether the image is displayed in a vertical or horizontal orientation. Another bit(s) of the reduced key hash result may be used to select the color of the image. Other bits may be used to indicate certain clearly identifiable features of the image or possible image types.

In another embodiment, the key verification technique may be used to generate different series of musical notes or chords. For example, certain bits are used to select the type of musical note and the remainder bits are used to generate duration, meter rate, octave change, etc.

Although these embodiments have described comparison of selected or generated items for users in close proximity and contemporaneous, it is contemplated that such comparison may be performed remotely (e.g., over telephone lines when comparing audible sounds or over television when comparing images). Such comparison may occur substantially contemporaneous or entirely non-contemporaneous when lesser data security is acceptable. The comparison is most effortlessly done when the two presentations are not just contemporaneous but simultaneous. Simultaneity might be prevented by substantial communication delays from sender to receiver. However, since the receiver is using a computed estimate of the sender's time, if the disagreement is one of simple offset (as would be the case

when the disagreement was due to transmission delay of the verification packet), the receiver's computation can include a manually controlled offset (perhaps seen by the user/operator as a knob or other continuous control) that can be adjusted to cancel any time difference between the two presentations. In the audio-based embodiment, the time difference might be perceived as an echo, for example, and the knob could be viewed as an echo cancellation control. This echo cancellation does not reduce the security of the key verification process.

While this invention has been described with reference to illustrative embodiments, this description is not intended to be construed in a limiting sense. Various modifications of the illustrative embodiments, as well as other embodiments of the invention, which are apparent to persons skilled in the art to which the invention pertains are deemed to lie within the spirit and scope of the invention.